II Sysops

A bi-monthly newsletter for, and by, Apple II sysops
Volume 1, Number 2 July/August 1991

Board-by-Board News

What's happening on bulletin boards around the world

Infinity's Edge BBS
(510) 820-9401 [CAOAK]

Paul Parkhurst, author of the Oggnet GBBS networking software, has announced ANSI-Term, a terminal program for the Apple IIgs which uses the SHR screen to display ANSI color graphics. It is expected to be released in October and will sell for $89.95. Features include: Auto-start Zmodem, scrollback, and a bunch of other stuff.

InSync Software

InSync has announced ProTERM 3.0, which includes a host of new features, including a text-based ANSI emulator. The press release made no mention of the reported problems with ProTERM's Zmodem.

L&L Productions
(303) 420-3568 [CODEN]

A few nasty complaints on the L&L Support BBS have resulted in a nasty response from Lance Taylor-Warren. In the system news, Lance says "I'd hate to take drastic measures that affect everyone, but ... I'll seriously consider taking this system down and/or revoking sysops right to support. Now I realize this will not be fair to those that have [been] very happy with support. But, I'm tired of it and it's up to YOU to put a stop to the few that will cause trouble for the rest... Unless sales for GBBS Pro pick up soon, we may all be out to dry!" He went on to ask GBBS users their opinion of going to an annual fee bases support setup, where you would have to pay for access to the support BBS.

The Captian's Quarters
(614) 294-0556 [OHCOL]

FutureNet has started up and currently has eleven nodes scattered across North America. FutureNet is similar to WWIVnet in that each networked conference is hosted by a particular node, and nodes can pick and choose which conferences they want to carry. FutureNet is part of the METAL / Future Vision package which sells for $85. FutureNet and Oggnet will soon be linking up via a gateway, and there has been talk of linking FutureNet to Fido, but no public action yet in that area.

StarPort BBS
(703) 931-0947 [DCWAS]

Joe Schober is testing version 2.0 of EBBS (may be available by the time you read this). EBBS runs under ModemWorks from Morgan Davis, and is network-compatible with ProLine. The software sells for $65 (not including ModemWorks), which includes a lifetime of free upgrades.

Niagara Software
(716) 689-7254

Niagara Software has just released a new BBS program called Vantage BBS. Unfortunately I didn't get a lot of info on this one, but a free demo was in the works last time I checked.

USRobotics
(708) 982-5092 [ILCHI]

US Robotics is running a special on their high-speed modems. Qualified sysops can get a Dual Standard for $399. Call their BBS for more info.

In This Issue...

Subscribe to *II Sysops*

Every other month, you can receive *II Sysops*, jam-packed with useful information, delivered to your mailbox, no matter where you live. Send your check or money order today to:

II Sysops
P.O. Box 720
Eldersburg, MD 21784

Payment must be in U.S. funds. Allow six to eight weeks for delivery. If possible, send us the mailing label from this issue of *II Sysops* with your payment. Thank you!

II Sysops Subscription Rates

	U.S.	Canada	International
1 issue	$2.50	$3.00	$3.75
3 issues	6.00	7.50	9.00
6 (1 year)	10.00	13.00	16.00
12 (2 yrs)	18.00	24.00	30.00

Have your friend subscribe! If you are a paid subscriber, send us the name of a friend who has never received a copy of *II Sysops*. We'll send him (or her) a free issue. If he (or she) subscribes, we'll add one issue to the end of **your** subscription, free of charge.

Advertise! Do you have something to sell to Apple II users? Promote it in the pages of *II Sysops*. Your ad is guarenteed to reach at least 200 II users. Ads must be submitted camera-ready, and can be black & white only. Rates are $50 for a full page, $25 for a half-page, $12.50 for a quarter page, and so on. Send us your ad with payment before the deadline and it will be included in the next issue. (Deadlines listed below.)

Write! Send in your submissions. We're looking for letters-to-the-editor and feature articles. Published letters earn a one-issue credit on your subscription and published features earn a one-year credit. Contact us before writing feature articles. (If you want to see promotion of your product in *II Sysops,* see the paragraph above about ads. If you just want your product mentioned, send us a product announcement and we'll mention it in Board-to-Board news.)

Deadlines: Issue 3 (Sept/Oct) submissions by Sept. 20. Issue 4 (Nov/Dec) by Oct. 29.

The Apple II and High Speed Modems
ACall For Action
Part 2

By Mike Garvey
Sysop of Valhalla BBS (415) 221-4370

The previous article discussed the problem of data-loss with high-speed modems and the Apple II and some theoretical hardware and software solutions to alleviate this problem. This article will begin to discuss the real-world issues involved in using a high-speed modem with existing Apple II hardware and software beginning with basic device interfacing.

Cable Confusion

The lowly serial cable, judging by the number of plaintive cries for help I've seen posted on BBSes over the years, has been blamed for everything from data-loss, erratic hardware and software behavior, crashed computers, and even global-warming (well, maybe not the latter). In reality the problems likely to arise from the cable are few and far between, and cabling is much simpler now than it was in the past.

The Apple II Sysop attempting to interface a high-speed modem need only consider a total of four cables, and two of these will give less than satisfactory results depending on the software used. It is not very surprising that the problem cables are the ones that are sold as 'standard' cables (i.e. Mac/IIGS modem and 25-pin 'straight-through' serial cables) -- RS-232 is less of a black-art now than in the past, but still, these general-purpose cables cannot possibly handle all serial interfacing projects. To properly interface two serial devices, one should be somewhat familiar with what the hardware AND software expects from the RS-232 interface and vice versa.

RS-232 is a standard for serial interfacing that specifies among other things, the uses of certain pins on a 25-pin connector such that any two RS-232 compatible devices can be reasonably expected to work properly when cabled together. The signals used by RS-232 serial ports and modems comprise a subset of the standard and include the following:

Pin-7: Circuit Common. Often mistakenly referred to as 'Signal Ground,' (GND) this pin has nothing to do with ground or earth. It acts as the absolute voltage reference for the interface circuitry, the point in the circuit from which all voltages are measured. Any two RS-232 devices *must* have this signal connected.

Pin-2: Transmitted Data (TxD), Pin-3: Received Data (RxD). These two pins carry the communicated information to and from the serial devices.

At this point, a brief mention of the terms Data Terminal Equipment (DTE) and Data Communications Equipment (DCE) is necessary. Rather than elaborate on the complete definitions, suffice it say that a DTE device transmits data on pin-2 and receives data on pin-3, while a DCE device transmits data on pin-3 and receives data on pin-2. While a modem is universally considered a DCE, a computer's serial port usually can be changed by hardware or software to behave like either a DTE or a DCE. RS-232 exists to interface DTEs and DCEs, and it is the responsibility of the interface (and therefore, the cable) to ultimately allow for this.

Pin-20: Data Terminal Ready (DTR), Pin-6: Data Set Ready (DSR). These two pins are used as the primary hardware handshaking signals. Hardware handshaking is used to determine if the two serial devices are ready to exchange data: whether they are powered-up and ready to go, whether a device's buffer is full and communications should be paused, etc.

Pin-4: Request to Send (RTS), Pin-5: Clear to Send (CTS). These two pins are used for various purposes including secondary hardware handshaking. The U.S. Robotics Courier HST modems use these signals to handle the modem's internal buffer.

Pin-8: Data Carrier Detect (DCD). Its uses vary, but on a DTE it is frequently used to disable data reception.

Apple II communications software and the HST modem usually use DTR (although the HST can be configured to ignore it and other RS-232 signals -- you don't want to do this) to actually control communications between the computer and modem. Typically DTR will be off when non-communications software is running and on during a communications session. Dropping DTR during a session usually signals the modem to end communications and hang-up. ProTERM and ACOS handle DTR in this way; the GS firmware behaves somewhat differently -- often toggling DTR on a warm or cold restart.

One of the curiosities of ACOS has always been its use of the DSR signal. In ACOS 1.x and 2.x, the default function of DSR is akin to that of the DCD signal, namely to signal to the software that a connection has been made between the remote and local modems and that data communications can occur. With ACOS 2.x, a Sysop has the option to override this setting and use DCD instead. ProTERM is happy with either arrangement.

As mentioned before, the HST uses CTS and RTS to control its internal buffer. When the buffer approaches capacity, the modem signals the computer to stop transmitting by lowering CTS. When the modem has sent enough data over the link to half empty the buffer, it signals the computer to resume transmitting by raising CTS. The HST responds to a lowered RTS by halting the flow of data to the computer. A computer could use the RTS signal to tell the modem to pause and resume the passing of data but no Apple II software does this, so RTS can be effectively ignored. This is a pity, because as I mentioned in the previous article, this is one way a slower computer can prevent data-loss at high speeds. Similarly, such hardware flow-control is absolutely essential for maximum throughput at high speeds.

Tying Things Together

The Apple II Super Serial Card (SSC) supports all of these signals and is configured as a DTE when the jumper block is in the MODEM position, as long as the seventh DIP-switch in row-1 (SW1-7) is closed (on). This enables DCD on pin-8. Additionally if SW2-7 is closed (on), a Secondary Clear to Send signal is available on pin-19. Since the HST doesn't use this pin, this signal is not necessary to our interface and SW2-7 should be left open (off). The SSC manual is very unclear about this and other important features of the card.

With ACOS's unique treatment of the DSR signal in mind, the following cable will provide a satisfactory interface:

Super Serial Card (DTE) (Male DB-25)	Modem (DCE) (Male DB-25)
2 (TxD)	2 (TxD)
3 (RxD)	3 (RxD)
4 (RTS)	4 (RTS)
5 (CTS)	5 (CTS)
6 (DSR)	6 (DSR)
7 (Common)	7 (Common)
8 (DCD)	
20 (DTR)	20 (DTR)

This is the cable that L&L Productions sells as 'GBBS Cable #3. The connections between DCD and DTR at the DTE, as well as RTS (which is unused, as I mentioned), prevent any electrical ambiguity and resultant erratic behavior, due to not connected but occasionally asserted signal-pairs.

I have been using ACOS with my HST since 1987 (I believe I was one of the first GBBS Sysops to own an HST), so I was obliged to use this cable with ACOS 1.2 and 1.3. Sysops using ACOS 2.x can also use the following cable, provided they choose the DCD carrier-detection option in CONFIG.SYSTEM:

Super Serial Card (DTE) (Male DB-25)	Modem (DCE) (Male DB-25)
2(TxD)	2 (TxD)
3 (RxD)	3 (RxD)
4 (RTS)	4 (RTS)
5 (CTS)	5 (CTS)
6 (DSR)	6 (DSR)
7 (Common)	7 (Common)
8 (DCD)	8 (DCD)
20 (DTR)	20 (DTR)

So with ACOS 2.x, a Sysop CAN use one of those 'standard' modem cables. I should also point out that while there is nothing wrong with using a cable with all twenty-five lines connected here, these cables are usually more expensive and gain you nothing in this application -- save your money and use the simpler cable.

The Apple IIGS modem-port functions as a DTE provided the 'Device Connected' setting in the Control Panel is set to 'Modem.' However, the GS modem-port does not use a standard RS-232 25-pin connector, nor is it internally wired as an RS-232 interface. It is in fact, an RS-422 port using an 8-pin mini-DIN connector.

Although RS-422 is logically identical to RS-232, its electrical standards are quite different. For example, on an RS-422 interface there are two transmitter lines -- one positive and one negative. There are also two receiver lines -- one positive and one negative. The important thing to note is that an RS-422 device can be connected quite easily to an RS-232 device. The 'standard' IIGS modem cable is wired this way:

GS Modem Port (DTE) (Male Mini DIN-8)	Modem (DCE) (Male DB-25)
1 (HSKo)	20 (DTR)
2 (HSKi)	8 (DCD)
3 (TxD-)	2 (TxD)
4 (Common)	7 (Common)
8 (RxD+)	
5 (RxD-)	3 (RxD)

The two signals HSKo and HSKi correspond approximately to the RS-232 DTR/DSR handshaking signals and the signed TxD/RxD pairs are also visible.

This cable doesn't support the CTS/ RTS pair that the HST uses for hardware flow-control. Additionally, the connection between pin-4 and pin-8 at the DTE is not really necessary and can cause problems with ACOS detecting and responding to carrier-loss (the BBS never realizes a caller has hung-up). Many Sysops can attest to this.

Paul Parkhurst (The Oggman) recommends the following cable for use with the GS modem-port and the HST:

GS Modem Port (DTE) (Male Mini DIN-8)	Modem (DCE) (Male DB-25)
1 (HSKo)	20 (DTR)
2 (HSKi)	8 (DCD)
3 (TxD-)	2 (TxD)
4 (Common)	7 (Common)
5 (RxD-)	3 (RxD)
7 (GPi)	5 (CTS)

Please turn to HIGH SPEED MODEMS, page 7...

Interview w/ Andy Nicholas

Andy Nicholas has become a highly respected figure in the Apple II Community. He first become widely known for SuperTAC, the now-standard transfer module for GBBS "Pro." He's made numerous other contributions, and has landed a job developing Finder GS for Apple. I had the chance to interview him recently, and this is what he had to say:

II Sysops: How did you first get into Apple II computers?

Andy Nicholas: I first started using an Apple][+ at a friend's house before I started in 9th grade. That summer I learned a little about AppleSoft BASIC and a lot about some neat Apple II games like Apple Panic (one of the first "hit" games that showed up on a microcomputer, and specifically for the Apple II). Later in 9th grade we had one Apple][+ with 48k that students could use -- I started writing AppleSoft programs that year. It was only towards the end of my sophomore year in high school that I kind of outgrew BASIC and started looking into assembly language. I started using Merlin and wrote a couple of small programs -- later as a high school project which lasted until I graduated I wrote a test grading system for Scantron card readers (you know, those machines that you feed the cards you have to mark with a #2 pencil through?). The teachers loved me for making it easy to give multiple choice 200 question exams -- the students were so grateful.

IIS: When did you get involved in BBS's?

Andy: I started getting involved in BBS'ing right around the end of my senior year in high school on a friend's system where I was co-sysop. I never ran my own BBS -- I was very happy to sit on the sidelines and try to help by making it the best it could be. I never had the time or patience to run the board day-in and day-out and make sure the equipment was working, etc. Those were the days when hard drives were rare; we originally ran the BBS from 6 floppy (5.25") drives.

IIS: Is your friend's BBS still up?

Andy: Nope, he took it down a couple of years ago.

IIS: Where did you go to high school?

Andy: I went to Lebanon High School in Lebanon, Pennsylvania. I grew up there and have a lot of fond memories of high school.

IIS: Your first contribution for sysops was SuperTAC, am I right?

Andy: Yes and no -- I also wrote the Xmodem and Ymodem drivers for GBBS. Gee, that was right after I graduated from high school (seems like such a long time now). SuperTac was the first thing I did which I thought was really neat other than my card-reader program back in high school. I did the Xmodem and Ymodem drivers for L&L Productions. I also wrote the original GS clock driver and GS serial port drivers that GBBS uses. The serial port driver uses the firmware. I learned a lot by writing (and rewriting and rewriting) those drivers.

IIS: Why did you write SuperTac?

Andy: I think I did it at the time just because there wasn't any kind of decent file transfer system that was simple to understand and easy to use. I had always liked the clean layout of Ascii's Express's file transfer system and had understood why AE:Tac had been developed, but also realized that at the time there was no simple, quick way to have a library of files on a BBS. I also hated quotas and stuff like that which is why SuperTac didn't have any kind of quota system (or credit system) through version 6.x.

IIS: Did you worry about it being used for pirating?

Andy: Not really. I knew it would get used for pirating. I wrote SuperTac for legitimate BBS Sysops, but many pirates found out that it suited their purposes excellently also. I'm not sure if this is all good or all bad, the real answer lies someplace in between.

IIS: Are you still involved in developing SuperTac?

Andy: Not really. Larry Hawkins took over development after I finished the first 6.x version of SuperTac. He kind of took it over by default because I did so much work with Larry on the previous versions (he'd report bugs and think of ways to fix them). In retrospect I should have asked Larry to take it over sooner than just leaving SuperTac stew for a while. Part of the problem with SuperTac is that by v6, it had grown to be something enormously complex. So much so that I almost couldn't get all of the bugs out of it without breaking something else. To a certain extent I'm kind of glad I didn't work on it any more because the point that I had left off from working on SuperTac was about the limit of what I was able to do with ACOS. Sure, some people can do a lot better, but I was getting frustrated and tired of working on it -- and college had a lot of work for me to do at the same time.

IIS: Why ShrinkIt, then? There were other Apple II packing programs before ShrinkIt.

Andy: I wrote ShrinkIt because I was convinced that the Apple II world needed a way to pack both files and disks in a better way that either BLU did for files or DDD did for disks. BLU uses static Huffman coding for its compression. It transmits the Huffman trees with the data and then decodes them on the other side. The code for the Huffman algorithms that BLU (and now ShrinkIt) was originally written by Richard Greenlaw. DDD on the other hand doesn't transmit any Huffman tree because it uses a pre-computed Huffman tree. It didn't yield best-case results, but it was fast. But, the problem with DDD is that it was primarily used by pirates and for that reason wasn't accepted by the mainstream Apple II community.

IIS: Did you expect anything in return for ShrinkIt?

Andy: The only thing I expected from ShrinkIt was that I wanted people to use it. To a certain extent I have to consider myself an artist, and like most artists I want people to see and enjoy my work. And, because I was also going to college I really didn't need money to keep living, so I could give the program away for free (which hopefully encouraged more people to use it).

IIS: Didn't you originally intend to charge for GSHK?

Andy: Yes, and once I found out that Andy Mcfadden's NuLIB program was going to be free, I thought we might actually lose money by charging for something which people could get for free, so GS-ShrinkIt was eventually made Freeware.

IIS: Any new features in store for ShrinkIt?

Andy: Maybe not many for 8-bit ShrinkIt, but for GS-ShrinkIt, yes. UnZipping is really in GSHK's future, as is the ability to extract files from StuffIt Deluxe, Compactor, and the new king of the hill for PC archivers, ARJ. GSHK v1.1 will use a large amount of resources also (v1.1 has been 80% done for about 9 months now, but the Finder has precluded my working on it at all).

IIS: What can we expect in the next version of Finder?

Andy: For one thing, the next version of the Finder makes Finder v1.3 pale by comparison. The Finder used to be about 60,000 lines of all 65816 assembly code. Now, it's many lines larger than that (I haven't actually checked to see how many source lines it is). You should probably bet on me trying to incorporate features currently unique to Macintosh Finder 7.0 into the GS's Finder.

IIS: Are you part of a team of people working on Finder?

Andy: No, it's just me. Dave Lyons (Toolbox Engineer) has helped out an enormous amount by helping me debug things and writing some significant sections of code, but I'm the only "Finder Engineer."

IIS: Where did you go to college?

Andy: I went to Moravian College in Bethlehem, Pennsylvania. I graduated in June of 1990 with a bachelor's degree in computer science. Before I graduated we were using a bunch of Sun SPARCStations for classwork. Having the workstations handy helped while I was developing GS-ShrinkIt because I had a working C source debugger that I could run archives through and see if the assembly code I had written was actually working (or not!)

IIS: Then you went to work for Apple? Did you apply there or did they seek you out?

Andy: Actually, I sent a bunch of resume's to Apple and hoped someone would listen. At Kansas City last year while Ralpha Russo was talking to the developers I was sitting outside being interviewed by Ron Lichty. I then interviewed "for real" in September in Cupertino and started work on November 5th.

IIS: Why Finder? Did you ask for it, or was there just an opening?

Andy: There was an opening for a Finder engineer, even though I was convinced that I was the man to work on the IIGS Toolbox. I couldn't have been more wrong. Tools weren't my bag at all. Finder has turned out to be more, uh, interesting, than working on the Toolbox ever would have been. I thought I could just go in and work on the IIGS toolbox -- it's nowhere near that simple.

IIS: When did you start working on Zmodem?

Andy: I started working on a 16-bit implementation of Zmodem back in January of 1990. I converted the 16-bit code to 8-bit for GBBS's Zmodem, and Proline's Zmodem.

IIS: What is the 16-bit Zmodem used with?

Andy: Nothing right now. I still have these comm programs which I never finished that have the 16-bit Zmodem code built into them. The 16-bit code is now much more primitive than the 8-bit code because I put much, much more work into the 8-bit drivers after I did the conversion. I don't think I'm going to do anything with the 16-bit code. If anything, the thing to do would be to take the 8-bit code which is now much more robust and port it back to 16-bit code.

IIS: Any plans for other implementations of Zmodem?

Andy: Not right now. But, because I realize I can't do it all, and I **want** people to USE Zmodem, I'm going to give copies of the Zmodem source code to each and every person who comes to my session at Kansas City. And it'll be free for that person to use however they see fit. The session is called "Zmodem, ShrinkIt, and a Surprise" The surprise is that I'm giving away the Zmodem source code. [Ed: Andy has since made the source code public domain and it is currently being adapted for METAL, ANSI-Term, and probably several other applications.]

IIS: Who would you say is your biggest role model, or "idol"?

Andy: Right now or at what time in my life? I don't have any role models right now -- maybe I need one. I had always heard about people at Apple's exploits like Jobs and Wozniak, but I didn't know them and didn't idolize them. Some people I met from Apple along the way I thought were really cool (like Jim Mensch, who currently works for Apple Macintosh Developer Support. He used to be the Lead Apple IIGS Toolbox Engineer) that kind of inspired me (until I got to know him; Mensch is weird). But right now I have a definite dearth of heroes. Hmm, maybe Tom Weishaar for a while.

IIS: How old are you?

Andy: I'm 23, being aged by the corporation faster than I want to be. Apple is (like any large company) rife with politics and intrigue, and just keeping your wits about you is very difficult and stressful. It seems like Apple is always in somekind of turmoil, and looking from the inside out gives you a very disturbing view of the goings on. I think that there are some interesting times ahead for Apple. I don't know how it will end, but the journey will be interesting, fer sure.

IIS: What's your favorite TV show?

Andy: I have two of them -- Star Trek: The Next Generation (you expected anything else?) and Quantum Leap.

IIS: ...favorite music group?

Andy: Yes (as in 90125, Drama, Time and a Word, Big Generator, Union, etc), Midnight Oil follows a close second.

IIS: What are your hobbies?

Andy: I go running, play a lot of ping-pong, and lift weights when I'm not getting razzed by the guys I lift with -- I lift weights with 2 guys from work, Bob Haven and Tim Swihart, and they are particularly fond of seeing if they can get me to laugh halfway through a lift (which causes me to, of course, get myself pinned to the bench).

IIS: You probably hear a lot about "the death of the Apple II". What do you think is in the future for the II line?

Andy: What do I think is the future of the II line? That's not a really fair question because being on the inside I KNOW what the future is, and if I were you I would take a good long look at what Apple has done so far and evaluate the future based on that. The future does not hold many surprises (unless you surprise easily!)

IIS: Okay, what do you think when people say "The II is dead."?

Andy: In comparison to other machines? Like the PC? Yes, the PC way, way, way outshadows the II line right now. Do I like this situation? No. Can I do anything about it? No. I'm an engineer that works for a big company, that's all.

IIS: What does your family think of your job?

Andy: They think it's neat -- my mom misses having me around the house (as I stayed at home during the 5 months after I graduated until I started at Apple), and my little sister thinks I'm going to get her a Ferrari for her birthday. (Eeek!) A large portion of my salary goes for rent -- I drive a Honda Civic, not a Ferrari.

GBBS "Pro"

First in a series of reviews of major Apple II BBS programs

"Which BBS program should I run?" This should be a familiar question. Everyone who has considered purchasing a program for their machine has asked this question when presented with multiple programs with the same function. It is always a question of tradeoffs, most often regarding what features are available at what cost.

The BBS world is much the same in this. There are now many choices available for systems to run, from many different sources. However, there are three that stand out from the rest. They are ProLine (from the Morgan Davis Group), GBBS "Pro" (from L&L Productions), and METAL (from Wilson Wares). Each has their features, and each their drawbacks. I'm not here to decide for you which is the best; I'm just here to let you know about each product. The decision on which to run is up to you.

Due to time constraints, this review will be presented in three parts, one per issue, dealing with, in turn, GBBS "Pro", ProLine, and METAL.

GBBS "Pro"

GBBS "Pro" is probably the Apple II BBS most familiar to users. There are GBBS systems spread far and wide, offering variety with a common base, uniqueness with familiarity. This is due to the core structure and ease of mutability of program, allowing a sysop to customize and personalize their system, while giving the sysop a good framework to build on.

This isn't to say that the framework itself is incomplete. As-is, GBBS "Pro" can be run without modification. It has what is a BBS kept in four small segments: LOGON.SEG, which handles getting new user accounts created, the login procedure for security, and getting the user's account information loaded into memory; MSG.SEG, which handles everything

HIGH SPEED MODEMS from page 4...

This cable takes advantage of the additional general purpose input (GPi) available on pin-7 of the GS modem-port (and certain Macintosh models) to support the CTS signal. Provided the 'DCD Handshake' setting is set to 'YES' in the Control Panel, the HST can signal the computer to stop transmitting data until it can empty its buffer. Without this ability, it is possible to overrun a slower (300-2400 bps) remote modem when oper- ating at high-speeds, with unfortunate results.

This cable is totally compatible with other Apple II communications software and is preferable to other flow-control cable configurations that sacrifice DTR by connecting HSKo to RTS and HSKi to CTS (with or without using the GPi signal). Software can still hang-up the modem by dropping DTR rather than having the modem hang-up (+++, followed by ATH) -- a much better solution, in my opinion, and hardware flow-control (trans- mitted data) is still supported.

In the next article, I will discuss configuring the SSC and the GS modem-port; setting the HST's DIP-switches, changing the modem's settings and non-volatile memory (NRAM); and setting up modem INIT-strings for various Apple II communications software.

regarding messages in the system, including the mail system and the public message areas; SYSTEM.SEG, which handles all the things a sysop needs to maintain the system, like editing/verifying users and files, adding new public message areas, and other security related options; MAIN.SEG, which links up all of the segments, giving access via menus, as well as other system features like a public text file area for permanent informational files, and a simple file transfer area.

All these segments are given in ACOS (All-purpose Communications Operating System -- GBBS' custom language) source code, which can be modified by anything that can modify text files, which ACOS can do itself. This gives the system a flexibility not often seen on systems, allowing you to make changes to the system as it runs, on all levels.

The language ACOS is designed to be used to run communication applications, specifically bulletin boards. The message files it uses to store a list of messages are proof of that. Messages are stored in a file, referenced by a list of pointers to their location, making deletion and renumbering of the messages a snap. When a KILL instruction is executed, the storage taken by the message is deallocated to be overwritten by a message to be stored later. Then the CRUNCH statement goes through the array of pointers, moving them down to cover up the blank spaces in the array, the practical upshot of which is that when you kill message 1 and crunch, message 2 takes message 1's place, 3 takes 2, etc., taking very little time.

Since messages are the primary function of a BBS, one is always concerned with the storage used up by the messages. ACOS' message files take action by reducing the size of messages using a packing scheme which is quick to decode, using the normally unused 8th bit. The only problem with this is that emulations that depend on 8-bit ASCII are difficult to store and retrieve.

Each message has a 2-byte integer that can be assigned to it, used like an integer array, which is used by GBBS as a "true-value counter" to discern what messages haven't been read. This same storage location is used as a pointer internally to the location of the message in the file, so that when an old message is killed, instead of messily copying entire message text down to fill in the hole, just the internal pointers are changed. This allows old messages to be scrolled off the message areas rather quickly.

GBBS "Pro" however is pretty bland as-is. The system just doesn't catch the eye of the user, and does have some errors which aren't apparent until certain things happen. Any sysop who has ever run a GBBS has encountered the problem of the system announcing that all messages are new. In fact, L&L Productions knows of this problem, but rather than fixing it, gives a segment called NEW.MSG.FIX.S to fix the symptoms, but not the disease. It "fixes" the problem by either setting all messages unread for all users, or setting all messages as already read for all users, which is not a good thing to put the users through either way.

The support from L&L Productions leaves much to be desired. Lance Taylor- Warren will not fix a bug unless you can track it down to what is exactly wrong, and even then may not get it fixed. He isn't on his system often, and the mailbox on the system fills up quickly. (As he runs a pretty stock system, all mail is stored in the same file.)

Therefore, it is good that all the segments are given in ACOS source code. ACOS itself is a form of BASIC, but without all those messy line numbers, so it is very easy to learn. Every command is documented in the manual, so what is new to you is explained, with examples of the syntax of every command. Even the virgin source is printed in the manual, so you can reference it whenever you have problems with a new routine, a kind of "Well, how did they do it?" guide.

Installation of the system is handled by a special configuration program packaged with the system, which is automatically run when you first boot the system disks. It takes you through each step of the system configuration, including defining the modem you use, what clock you have installed, if you have a printer, and what kind of display system you have for different systems from an Apple][+ to the Apple IIgs. It also lets you define what form of storage you wish to install the system, be it on multiple 5.25" drives or other, larger capacity drives. The documentation even gives some information on certain memory addresses for you to use in making your own video, printer, and modem drivers, and external binary programs which are used similarly to the ampersand (&) commands in AppleSoft BASIC, which can be used to install alternate editors or enhance the system to provide for simulated arrays (which are not present in ACOS).

Extensions to the BBS are available from many sources. Many sysops who run their software with file capability have an area dedicated to GBBS/ACOS files, accessible by proving that your system is registered, which is checkable with the L&L Support BBS, where other extensions are available for download, including new versions of the system. Some of the most common additions to a system include enhanced file transfer systems, mail systems which use separate files for each user, and the capability to display PSE (ProTERM Special Emulation -- a way to send and display MouseText) characters and sound on the system console, which is an aid to designing special interfaces for Apple ProTERM users.

I find GBBS "Pro" to be a system more geared toward the sysop who likes to make changes and mold a system to reflect the sysop's personality. As a "set it up and leave" system, it isn't. GBBS needs attention, tweaking, and modifying. It can be put online immediately after purchase, but I can't recommend doing so. Look around at other systems, see what they have done with theirs, and then dive into making changes. Get the system looking and feeling the way that you are comfortable with, and then put it online for others to use. This will increase your possibility of early success, and save your users many headaches encountered had you made major changes to it while they used it.